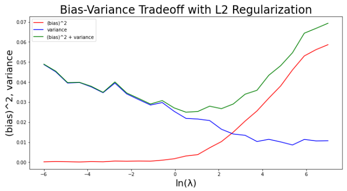
Introduction to Machine Learning

Introduction to Regularization

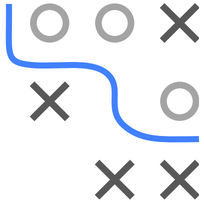


Learning goals

- Understand why overfitting happens
- Know how overfitting can be avoided
- Know regularized empirical risk minimization

WHAT IS REGULARIZATION?

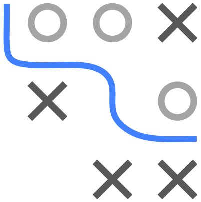
Regularization comprises all methods that add preferences for specific solutions (**inductive bias**) to a model, usually in the context of “low complexity” priors (shrinkage and sparsity). By controlling complexity we can reduce overfitting and achieve an optimal bias-variance tradeoff.



- **Explicit regularization** methods define an explicit measure of model complexity and add this as penalty to empirical risk (e.g., $L1/L2$)
- **Implicit regularization** includes removing outliers, early stopping, data augmentation, parameter sharing, dropout or ensembling
- **Structured regularization** methods incorporate structural prior knowledge over groups of parameters or subnetworks (e.g., the group lasso ▸ Yuan and Lin, 2005)

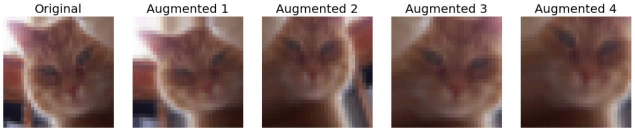
REGULARIZATION FOR INVARIANCE

Prior knowledge can also be of the form that predictions should remain invariant under certain input transformations.
In image classification, label "cat" should hold regardless of position or size of relevant object (translation/scale invariance)



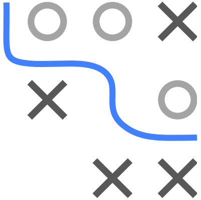
1. **Pre-processing**: By computing invariant features under transformations, downstream models too will respect invariances

2. **Data augmentation**: Extend training data by replicating inputs under invariant transformations (e.g., flipping/rotating images)



3. **Network architecture**: Build invariance property directly into network structure, e.g. CNNs ▸ Geometric DL (Bronstein et al., 2021)

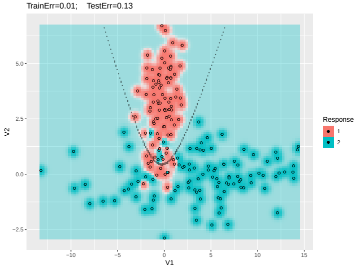
RECAP: OVERFITTING



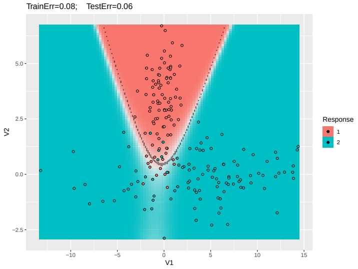
Reducing overfitting is an important application of regularization, so let's first motivate it from that perspective.

- Overfitting occurs when the model reflects noise or artifacts in training data which do not generalize (small train error, at cost of test high error)
- Hence, predictions of overfitting models cannot be trusted to generalize beyond the training data

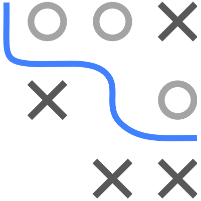
Overfitted model



Appropriate model



EXAMPLE I: OVERFITTING



- Assume we want to predict the daily maximum **ozone level** in LA given a data set containing 50 observations.
- The data set contains 12 features describing time conditions (e.g., weekday, month), the weather (e.g., temperature at different weather stations, humidity, wind speed) or geographic variables (e.g., the pressure gradient).
- We fit a linear regression model using **all** of the features

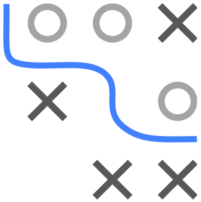
$$f(\mathbf{x} \mid \boldsymbol{\theta}) = \boldsymbol{\theta}^T\mathbf{x} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_{12} x_{12}$$

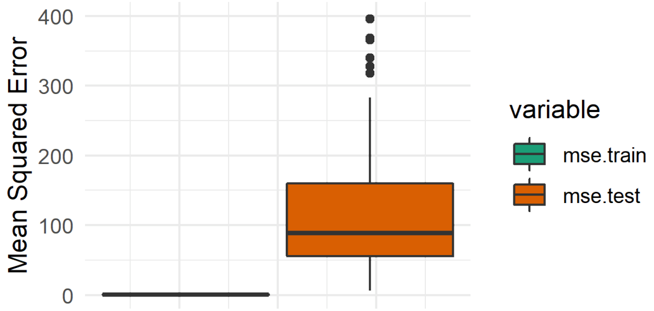
  with the $L2$ loss.
- We evaluate the performance with 10 times 10-fold CV.

We use (a subset of) the `Ozone` data set from the `mlbench` package. This way, we artificially create a “high-dimensional” dataset by reducing the number of observations drastically while keeping the number of features fixed.

EXAMPLE I: OVERFITTING

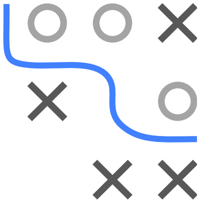
While our model fits the training data almost perfectly (left), it generalizes poorly to new test data (right). We overfitted.





EXAMPLE II: OVERFITTING

We train a shallow neural network with one hidden layer and 100 hidden units as well as a SVM with RBF kernel (and $C = 1e6, \gamma = 10$) on a small regression task. No form of explicit regularization is imposed on the models.

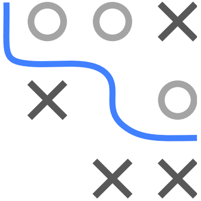


	Neural Network	SVM
Training MSE	0.00	24.88
Test MSE	0.72	86.15

- Both neural network and SVM perform significantly better on the training set
- The shallow NN even achieves zero training error (interpolating)
- Test error is significantly higher for both models, indicating overfitting

AVOID OVERFITTING

Why can **overfitting** happen in practice? And how to avoid it?

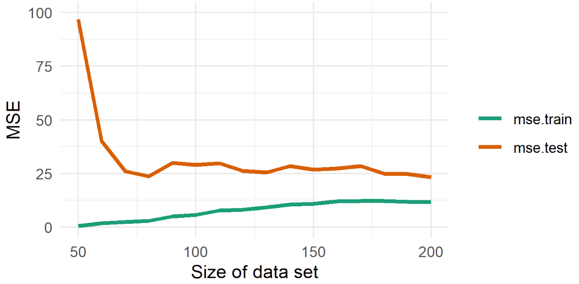


1. Not enough data
   $\rightarrow$ collect **more data**
2. Data is noisy
   $\rightarrow$ collect **better data** (reduce noise)
3. Models are too complex
   $\rightarrow$ use **less complex models**
4. Aggressive loss optimization
   $\rightarrow$ **optimize less**

AVOID OVERFITTING

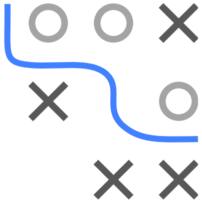
Approach 1: Collect more data

We explore our results for increased dataset size by 10 times 10-fold CV. The fit worsens slightly, but the test error decreases.



Good insight, but getting more data is often not feasible in practice.

AVOID OVERFITTING



Approach 3: Reduce complexity

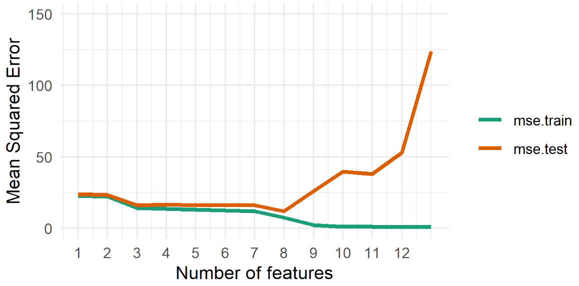
We try the simplest model we can think of: the constant model. For the *L*2 loss, the optimal constant model is the empirical mean

$$f(\mathbf{x} \mid \theta) = \frac{1}{n} \sum_{i=1}^{n} y^{(i)}$$

We then increase the complexity of the model step-by-step by adding one feature at a time.

AVOID OVERFITTING

We can control the complexity of the model by including/excluding features. We can try out all feature combinations and investigate the model fit.



Note: For simplicity, we added the features in one specific (clever) order, so we cheated a bit. Also note there are $2^{12} = 4096$ potential feature combinations.

AVOID OVERFITTING

Approach 4: Optimize less

Now we use polynomial regression with temperature as the only feature to predict the ozone level, i.e.,

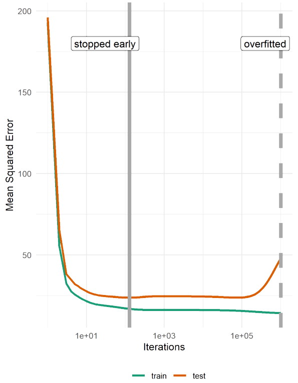
$$f(\mathbf{x} \mid \boldsymbol{\theta}) = \sum_{i=0}^{d} \theta_i (x_T)^i.$$

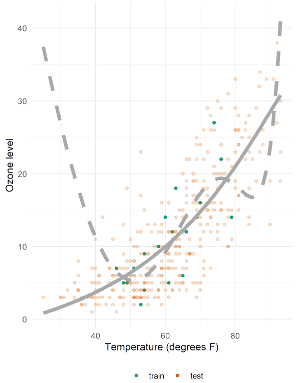
We choose $d = 15$, for which we get a very flexible model, which can be prone to overfitting for small data sets.

In this example, we don't solve for $\hat{\theta}$ directly, but instead, we use the gradient descent algorithm to find $\hat{\theta}$ stepwise.

AVOID OVERFITTING

We want to stop the optimization early when the generalization error starts to degrade.





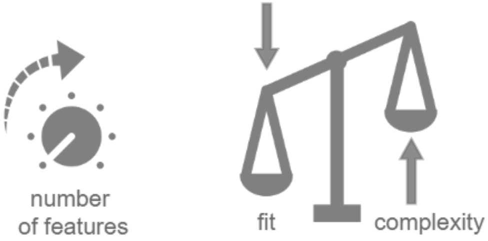
Note: For polynomial regression, gradient descent usually needs many iterations before it starts to overfit. Hence a very small training set was chosen to accelerate this effect.

AVOID OVERFITTING

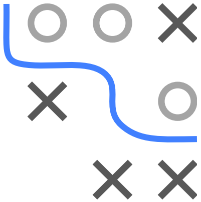
We have contradictory goals

- **maximizing the fit** (minimizing the train loss)
- **minimizing the complexity** of the model.

We need to find the “sweet spot”.

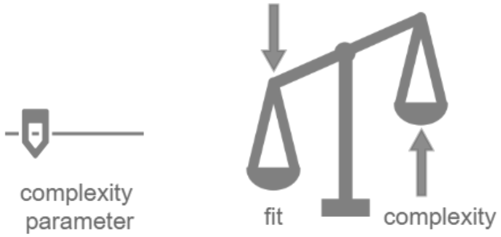


AVOID OVERFITTING

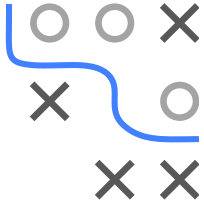


Until now, we can either include or exclude features in a binary fashsion.

Instead of controlling the complexity in a discrete way by specifying the number of features, we might prefer to control the complexity **on a continuum** from simple to complex.



REGULARIZED EMPIRICAL RISK MINIMIZATION



Recall, empirical risk minimization with a complex hypothesis set tends to overfit. A major tool for handling overfitting is **regularization**.

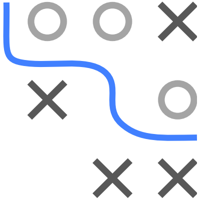
In the broadest sense, regularization refers to any modification made to a learning algorithm that is intended to reduce its generalization error but not its training error.

Explicitly or implicitly, such modifications represent the preferences we have regarding the elements of the hypothesis set.

REGULARIZED EMPIRICAL RISK MINIMIZATION

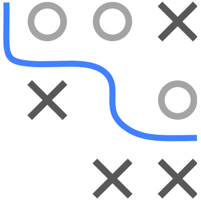
Commonly, regularization takes the following form:

$$\mathcal{R}_{\text{reg}}(f) = \mathcal{R}_{\text{emp}}(f) + \lambda \cdot J(f) = \sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right) + \lambda \cdot J(f)$$



- $J(f)$ is called **complexity penalty**, **roughness penalty** or **regularizer**.
- $\lambda > 0$ is called **complexity control** parameter.
- It measures the “complexity” of a model and penalizes it in the fit.
- As for $\mathcal{R}_{\text{emp}}$, often $\mathcal{R}_{\text{reg}}$ and J are defined on $\boldsymbol{\theta}$ instead of f, so $\mathcal{R}_{\text{reg}}(\boldsymbol{\theta}) = \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) + \lambda \cdot J(\boldsymbol{\theta})$.

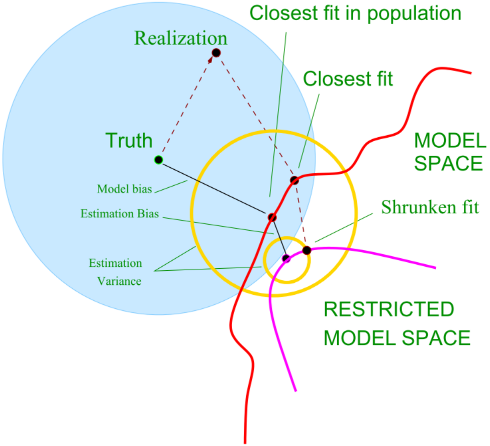
REGULARIZED EMPIRICAL RISK MINIMIZATION



Remarks:

- Note that we now face an optimization problem with two criteria:
  1. models should fit well (low empirical risk),
  2. but not be too complex (low $J(f)$).
- We decide to combine the two in a weighted sum and to control the trade-off via the complexity control parameter λ.
- λ is hard to set manually and is usually selected via cross-validation (see later).
- $\lambda = 0$: The regularized risk $\mathcal{R}_{\text{reg}}(f)$ reduces to the simple empirical $\mathcal{R}_{\text{emp}}(f)$.
- If λ goes to infinity, we stop caring about the loss/fit and models become as “simple” as possible.

REGULARIZED EMPIRICAL RISK MINIMIZATION



Hastie, The Elements of Statistical Learning, 2009 (p. 225)